CS290 How-To Outline

Overview from my perspective

The purpose of this NASA API is to serve up an overview of the changes to the huge amount of information which is kept by NASA. The API only returns a JSON object which can be traversed with different methods both in php and javascript.

1. What is the API.
   1. What data does it contain? It contains data which explains what has changed a crossed NASA’s different data repositories.
   2. Why is this important? It is important because there is so much data trying to keep track of all of the changes in a meaningful way would mean you would be flooded with so much information as to what has changed in the databases it would not be of any use.
   3. So we can use the provided functions to access the changes to the data in certain ways.
   4. We can then display this data by parsing the JSON objects into their component parts and displaying these apart with AJAX calls.
   5. Include a link to JSON and AJAX tutorials.
      1. I will refer to W3C schools with a qualification as to the quality of the tutorial.
      2. Include a link to HTML Dog tutorials with my preference for this tutorial page.
   6. We could also us PHP and this will be discussed if I have time.
2. What are the different slugs.
   1. The slugs correspond to a specific nasa category. It seems to be a more formal approach than using the category field as a slug might be a last name of the individual responsible for the data.
   2. The slugs id and description and parent all make up the categories data.
3. Go over the available methods.
   1. get\_recent\_datasets
      1. Required arguments: none.
      2. Optional arguments: count. Returns the amount of recent datasets.
   2. get\_dataset
      1. Required arguments: id, slug.
      2. Optional arguments: none.
      3. This can be used with both of these arguments. No additional arguments can be used except for [dev=1].
   3. get\_date\_datasets
      1. Required arguments: Date in the format yyyy or YYY-MM. Non – numeric characters are removed.
      2. Optional arguments: count for the amount of datasets to be returned.
   4. get\_category\_datasets
      1. Required arguments: id, set to the cateogory’s id. Slug, set to the category’s slug
      2. Optional arguments: count
   5. get\_tag\_datasets
   6. get\_search\_results
      1. Required: search. You set this with a string and you can use the + sign to combined your different words.
      2. Optional: count
   7. get\_date\_index: This provides an array of dates to the permalinks to the data. This can help define the tree structure, whatever that is. It also seems to have a count of how many permalinks from each date there are.
   8. get\_category\_index
   9. get\_recent\_datasets
4. Explain what the data is in the JSON response.
   1. “status” : “ok”, or not ok.
   2. A NASA “id” : this is a numerical id to each set of data
   3. A “slug” : This is the identification of the data used by nasa. It has all capital letters removed from what would otherwise be the title of the data.
   4. “title” : a title of the
   5. “description” :
   6. “post\_count” :
5. An example of how we would use each method.
   1. The methods are called with the GET method.
   2. Do this with little code excerpts.
   3. You can use the variable [dev=1] to have the output be formatted in a more human readable form in the browser.
   4. Explain how to use the browser dev tools. In other words F12. I will use Google Chrome as my examples for the keystrokes.
      1. How can I use the network connection to review the data I am receiving from NASA.gov. This probably shouldn’t be very long portion of the page.
6. Instead of an example of each case, I could provide a finished small formatted example
7. Refer to the JSON-API WordPress Plugin which NASA use to create their API.

* Overview Introduce the topic of the lecture pages and how it is organized.
* introduce the different method
* Define the different methods with both what an error looks like and what a successful return looks like. Picture examples, images maybe the actual text return. I like the idea of a screen capture of the return.
* Have a section with a button which causes an ajax request.
  + This will be a DIV which is originally has the request url, and then underneath it has the return data formatted with the dev tools. Perhaps this is just a couple of fields which build up the string to show how the arguments should be passed.
* I need to show how to parse the JSON information. Have a link to the for each loop Javascript loop.